Turnpike Problem

Naveen Belkale

Topics covered in presentation

- Introduction
- Combinatorial approach
- Polynomial approach
  - Berlekemp's algorithm
- Special case of distinct distances

Introduction

- Identifying point sets that realize pairwise distance multisets
- Applications
  - DNA sequencing (Partial Digest problem)
  - X-Ray crystallography
- Combinatorial solution in exponential time
- Polynomial factorization in polynomial time

Exponential solution

- Pyramid visualization
-

0 1 2 3 4 5

D_{05}

D_{04} D_{15}

D_{03} D_{14} D_{25}

D_{02} D_{13} D_{24} D_{35}

D_{01} D_{12} D_{23} D_{34} D_{45}

$D_{ij} + D_{kl} = D_{il} + D_{kj}$, $i \le k \le l \le j$

Sum of distances in kth row =
Sum of distances in (n-k)th row

Polynomial Solution

- Converting the problem into polynomial form
- Generating function associated with point set
  - $P(x) = \sum_i x^{a_i}$
- Distance generating function
  - $Q(x) = P(x)P(1/x)$
  - $Q(x) = n + \sum_i (x^{d_i} + x^{-d_i})$
  - $Q(x) = \prod_i P_i(x) P_i(1/x) R(x) R(1/x)$
  - $P_s(x) = \prod_{i \in s} P_i(x) \prod_{i \in \bar{s}} P_i(1/x) R(x)$

Polynomial solution...

- Berlekamp's algorithm for polynomial factorization.
- Polynomial solution is polynomial in order of maximum degree – O(n^3 + prn^2)
- Factorization coefficients module prime p
- Distinct degree factorization method
- Hensel's lemma to extend factorization modulo p to factorization modulo p^e

Berlekemp's algorithm

- Factoring modulo p
- u(x) is a monic polynomial
- u(x) made square free using differentiation
  - gcd(u(x), u'(x)) = v(x) gives power factors
- If u'(x) = 0, then u(x) = v(x^p) = (v(x))^p
- gcd(u(x), v(x)) using Euclid's method
- Therefore, u(x) = p1(x) p2(x) .... pr(x)

Key to the solution, v(x)

- Chinese remainder algorithm
- $$\text{if } (s_1, s_2, \ldots, s_r) \text{ is any } r\text{-tuple of integers mod } p, \\ \text{there is a unique polynomial } v(x) \text{ such that}$$
- $$v(x) \equiv s_1 (\text{modulo}(p_1(x))), \ \ldots\ , \ v(x) \equiv s_r (\text{modulo}(p_r(x))), \\ \deg(v) < \deg(p_1) + \deg(p_2) + \ldots + \deg(p_r) = \deg(u)$$
- gcd(u(x),v(x)-s1) divisible by p1(x) but not by p2(x)
- $v(x)^p \equiv v(x) \ (\text{modulo}\, u(x)), \ \deg(v) < \deg(u)$
- $x^p - x \equiv (x-0)\ (x-1)\ \ldots\ (x-(p-1))\ (\text{modulo}\, p)$
- $v(x)^p - v(x) \equiv (v(x)-0)\ (v(x)-1)\ \ldots\ (v(x)-(p-1))$

How to find the key??

Let $\deg(u) = n$; we can construct nxn matrix

$$Q = \begin{pmatrix} q_{0,0} & q_{0,1} & \cdots & q_{0,n-1} \\ \vdots & \vdots & & \vdots \\ q_{n-1,0} & q_{n-1,1} & & q_{n-1,n-1} \end{pmatrix}$$

where $x^{pk} \equiv q_{k,n-1}x^{n-1} + \ldots + q_{k,1}x + q_{k,0}\ (\text{modulo}\, u(x))$

Then $v(x) = u_{n-1}x^{n-1} + \ldots v_1 x + v_0$ is a solution if and only if

$$(v_0, v_1, \ldots., v_{n-1})Q = (v_0, v_1, \ldots., v_{n-1});$$

for the latter equation holds if and only if

$$v(x) = \sum_j v_j x^j = \sum_j \sum_k v_k q_{k,j} x^j \equiv \sum_k v_k x^{pk} = v(x^p) \equiv v(x)^p (\text{modulo}\, u(x))$$

Generating Q

if $u(x) = x^n + u_{n-1} x^{n-1} + \ldots + u_1 x + u_0$

and if

$x^k \equiv a_{k,n-1} x^{n-1} + \ldots + a_{k,1} x + a_{k,0} (\text{modulo}\, u(x)),$

then,

$x^{k+1} \equiv a_{k,n-1} x^n + \ldots + a_{k,1} x^2 + a_{k,0} x$

$x^{k+1} \equiv a_{k,n-1} (-u_{n-1} x^n - 1 - \ldots - u_1 x - u_0) + a_{k,n-2} x^{n-1} + \ldots + a_{k,0} x$

$x^{k+1} \equiv a_{k+1,n-1} x^{n-1} + \ldots + a_{k+1,1} x + a_{k+1,0}$

where,

$$a_{k+1,j} = a_{k,j-1} - a_{k,n-1} u_j$$

Berlekemp factoring algorithm

1. Ensure that u(x) is squarefree
2. Form the matrix Q
3. Triangularize the matrix Q – I, where I is nxn identity matrix finding its rank n-r and finding linearly independent vectors v1, ..., vr such that vj (Q – I) = (0,0,...,0) for 1<=j<=r
4. Canculate gcd(u(x), v2(x)-s) for 0<=s<p. The result will be nontrivial factorization of u(x)

- If the use of v2(x) does not succeed in splitting u(x) into r factors further factors can be obtained by calculating gcd(vk(x) – s, w(x)) for 0<=s<p and all factors w(x) found so far

Null space algorithm

- For finding v1,v2,...vr from Q – I
- In the kth row of Q matrix, using one column j make all other columns zero
- Repeat the procedure for all rows
- End when no unused column left in a row
- $$v^{[r]} = (v_{0,} v_{1,} \ldots, v_{n-1})$$
  $$\text{defined by the rule}$$
  $$v_j = \begin{Bmatrix} a_{ks}, \text{if } c_s = j \geq 0; \\ 1, \text{if } j = k; \\ 0, \text{otherwise.} \end{Bmatrix}$$

Factorization example using Berlekemp Algorithm

$$u(x) = x^8 + x^6 + 10x^4 + 10x^3 + 8x^2 + 2x + 8 \ (\text{modulo } 13)$$

$$Q - I = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 7 & 11 & 10 & 12 & 5 & 11 \\ 3 & 6 & 3 & 3 & 0 & 4 & 7 & 2 \\ 4 & 3 & 6 & 4 & 1 & 6 & 2 & 3 \\ 2 & 11 & 8 & 8 & 2 & 1 & 3 & 11 \\ 6 & 11 & 8 & 6 & 2 & 6 & 10 & 9 \\ 5 & 11 & 7 & 10 & 0 & 11 & 6 & 2 \\ 3 & 3 & 12 & 5 & 0 & 11 & 9 & 11 \end{pmatrix}$$

$$v^{[1]} = (1, 0, 0, 0, 0, 0, 0, 0)$$
$$v^{[2]} = (0, 5, 5, 0, 9, 5, 1, 0)$$
$$v^{[3]} = (0, 9, 11, 9, 10, 12, 0, 1)$$

$$u(x) = (x^4 + 2x^3 + 3x^2 + 4x + 6)$$
$$(x^3 + 8x^2 + 4x + 12)(x + 3)$$

Final result of matrix operations

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 12 & 0 & 0 \\ 0 & 0 & 0 & 0 & 12 & 0 & 0 & 0 \\ 0 & 12 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12 \\ 12 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 5 & 5 & 0 & 9 \\ 12 & 9 & 0 & 0 & 11 & 9 & 0 & 10 \end{pmatrix}$$

Special case of distinct distances

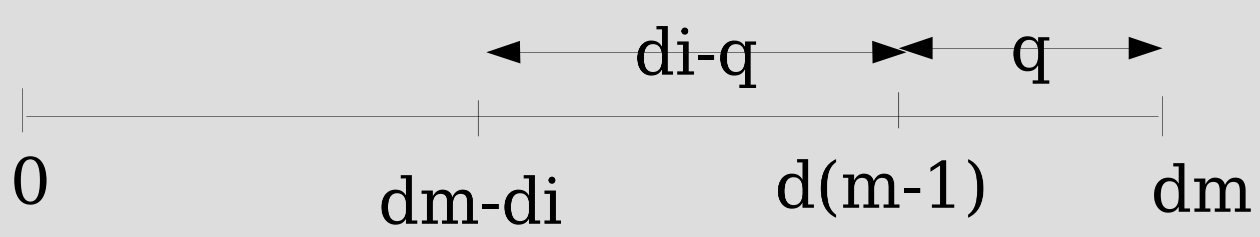
- Problem : when all the interpoint distances are distinct find the point set in polynomial time
- Solved using modification of the backtracking method
- Solvable in $O(n^2 \log n)$ time

Where do I begin?

- Let D = {d1, d2, ... , dm} be distance set
- Initialization
  - q = dm – d(m-1)
  - P = { 0, d(m-1), dm }
  - D = D \ { dm, d(m-1), q}

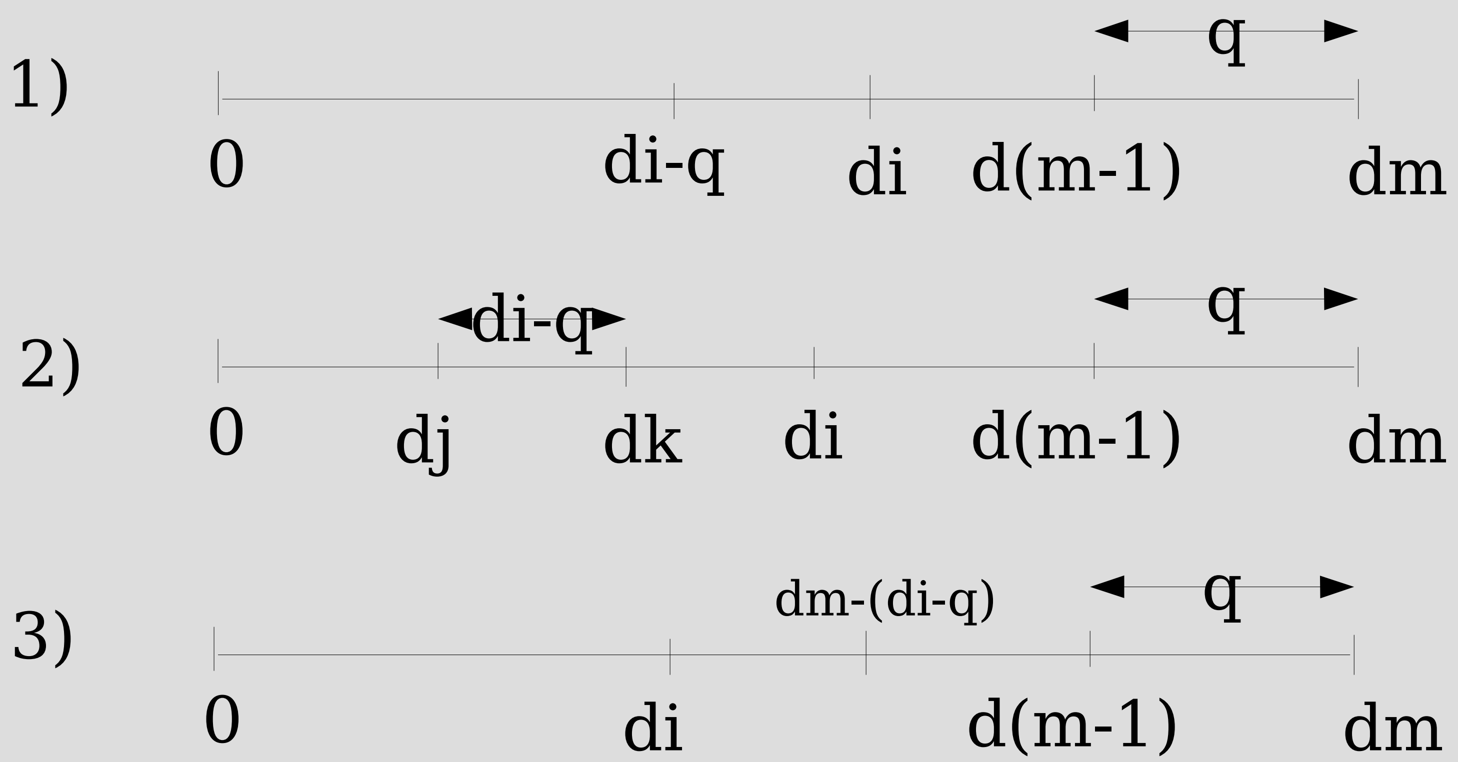
Left or right?

- When do we have a choice?
  - If di, dm – di, d(m-1) – di, di – q exist
- What to do when we have a choice?
- Save the state (D,P)
- Choose the point as dm – di
  - Distances covered di, dm – di, di – q
- What does this imply?



What if the decision is wrong?

- One point is at di. What can we say about di – q ?



So what did we achieve?

- No need to save the state in stack once a decision is wrong in the future flow
- In case we made one more wrong decision at dj, then correct point at dj and d(m-(dj-q))
- Then,
  - Distance (dm-(dj-q)) – (dm-(di-q)) = di-dj is the interpoint distance between di and dj!!
- This cannot happen and hence no need to save the stack
- Hence at each depth atmost one stack state is saved

References

- Skiena, Smith and Lemke: Reconstructing sets from interpoint distances
- Knuth: Art of computer programming -vol2
- Mangesh Gupte: ME project report on Turnpike reconstruction problem

Thank You